ON DC CONSTRAINED CODES FOR FIBER OPTIC APPLICATIONS

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

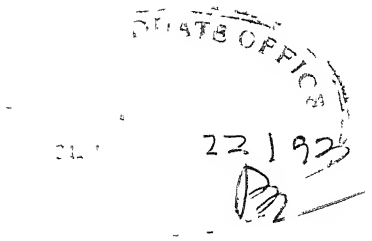
by

T. RAMESH CHANDRA

to the

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

January, 1993



CERTIFICATE

It is certified that the work contained in the thesis entitled "On DC Constrained Codes for Fiber Optic Applications" by T. Ramesh Chandra has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

January, 1993

P.K. Chatterjee
Professor
Department of Electrical Engineering
Indian Institute of Technology
Kanpur - 208016

EE-1993-M-CHA-CN

ABSTRACT

Line Codes which incorporate Error Control capability for highspeed fiber optic links, have been studied in this work. Balanced codes, which contain equally many 1's and 0's, suitable for transmission where dc free pulse formats and low complexity encoder-decoder implementations are required, have been studied. A class of block coset codes derived by partitioning linear block codes are discussed.

Balanced codes without error correction, derived from linear block codes are presented Balanced codes with Error Correction derived by partitioning a set of balanced words are described. Encoding and decoding techniques for such codes are developed. An improvement over the one dimensional codes, to provide burst error correction is presented. The Error Correction Capability of such array codes is discussed and the encoding and decoding algorithms are presented.

A comparison of the directly coded 8B10B block code with that obtained by combining two smaller length block codes is discussed. The features like run length, Running Digital Sum etc. for these codes are tabulated.The concept of guided scrambling and its use in line coding is also studied.

to
My Parents

ACKNOWLEDGEMENTS

I am grateful and indebted to Dr P.K Chatterjee for offering me this interesting topic to work towards my M.Tech. thesis. His suggestions and comments in many sessions we had, helped me very much to puruse the work with verve and enthusiasm.

I will long remember my stay at IIT Kanpur, in the company of Madava Reddy, Barari, Ranganadh and others with whom I had a happy time.

Finally, I would like to thank Chaitanya Babu (R.C.) for the enormous help he had done to me without any scruples.

RAMESH

CONTENTS

CHAPTER - 1

INTRODUCTION

In fibre optic communication systems, like in other digital transmission systems, the channel often does not pass dc. This causes the problem of baseline wander. One way to overcome this difficulty is to restrict the dc content in the signal stream using suitably designed codes. As a result, many codes having dc constrained property have been studied [14] The coding requirement is defined in terms of constraint on the Running Digital Sum (RDS) of the coded signal stream and the efficiency of such a dc constrained code is related to RDS in a definite way. The question of what is the best possible way to constrain the RDS and design a dc constrained code has been addressed by several authors and in the process, codes with error detection have been designed. Improvements have been done in terms of impelementation and reduced complexity.

Various mBnB codes are examples of such d.c. constrained codes which have found considerable application in fibre optic communication links.

1.1 Features of Line Codes

Line coding is the process of modifying a source signal to facilitate proper signal reception in the presence of transmission impairments. In fiber systems, following line code features are required.

(1) Bit sequence independence : The linecoder must adequately encode any source sequence.

(2) Small low frequency content : The transmitted signal should be balanced to allow for ac coupling in the receiver

(3) Transmission of adequate timing information : The encoded bit stream must contain enough level transitions to allow for proper operation of clock recovery circuitry.

(4) High efficiency : To keep noise bandwidth and terminal circuitry as low as possible, introduced redundancy should be kept at a minimum.

(5) Low error multiplication : An error which occurs during transmission should not result in many decoded errors.

(6) Low systematic Jitter : The sequence of transmitted bits must ensure that pattern dependent Jitter is kept low.

Keeping the above properties in mind, work has been done in designing mBnB block codes [4], which incorporate the above features.

1.2 mBnB Block Codes

The mBmB codes are fixed length binry block codes. Input data is divided into blocks of length m bits and each block is coded into a n bit codeword (n > m). This process increases the code rate, but, at the sametime, provides some advantages. The efficiency of the mBnB code is m/n For purposes of synchronization, the block duration of m bits is made equal to codeword duration n bits. Hence there is an increase in the code rate,

equal to n/m. Thetransmitted symbol bit duration $T_n = T_m.m/n$, where T_m is the bit duration of m bit uncoded block.

To limit channel data rate higher values of m and n are used which lead to higher efficiency codes. Usually, n is taken to be equal to m+1. Two alternate set of codewords with opposite disparity are used. This keeps a check on the Digital Sum Variation (DSV) of the code sequence.

The important features of these mBnB codes are :

(1) As m increases, the efficiency of the code increases. The increase in data rate and Bandwidth is smaller with larger value of m. With a large value of m, the coding complexity increases. But, for higher capacity systems, coding costs are less compared with total cost

(2) The mBnB codes exhibit error extension effect. For each bit of receiver decision error, whole block is in error and hence maximum error spread is m bits per bit of receiver error.

(3) With a large value of m, the number of consecutive similar bits (Run length) also increases and hence the code carries less timing information. The DSV also increases with increasing m.

(4) Block Coders use lookup tables for purposes of encoding and decoding. Thus the value of m has to be chosen in such a way that lookup tables can be implemented in an easy manner and availability of EPROM's at the required data rates is not a problem.

1.3 DmBIM and mBIC Codes

A new linecode called "DmBIM" which is suitable for very high speed optical digital transmission has been developed [14]. This code pocesses a good balance of marks and spaces. In this encoding process, the speed of input signal is increased by m+1/m times and then a mark bit is introduced in the (M+1)th slot. Mark inserted signal (Q) is converted to the DmBIM code(S) by the equation $S_k = S_{k-1} \oplus Q_k$ where, S_k and Q_k denote the kth signalbit of S and Q, respectively. In the decoding process Q is obtained from, $Q_k = S_k \oplus S_{k-1}$ where $\oplus$ represents mod-2 addition. The original information signal is then recovered through m/m+1 speed conversion.

The (m+1)th bit is complement of the mth bit, which is generated automatically by the encoding procedure. The increase in code rate is m+1/m. The Run length of this code is m+1. The spectral characteristics of the DmBIM code show that there is no discrete component except dc component. As block length m increases, suppression of high frequency and low frequency components weakens and the spectrum assumes the shape of a random signal.

Another line code which can be used for high speed transmission system, is the mBIC code [14]. In this coding format, an additional bit is introduced, after every m bits of the input stream which is the complement of the mth bit. Thus, the maximum run length is m+1. The received mBIM coded bit stream is synchronized by the frame pulse. The stream then has its

transmission speed converted to the original information speed by a speed converter, after errors in the stream have been detected by the c bit. Spectral characteristics of the mBIC show that the high frequency and low frequency components are suppressed.

The mBIC codes exhibit the following features :

(1) They suppress Jitter more effectively when a scrambler precedes the coder.

(2) Error rate characteristics in a transmission system which suffers from intersymbol interference are improved by use of mBIC code.

An error detection method using c bit checking proved very effective in this case.

1.4 Scrambling of Data Stream

Scrambling is another technique which has the capability to meet some of features of the line coder mentioned earlier. Self synchronizing scrambling is one where, the source sequence is multiplied mode 2 by x^d and then divided by d(x), where d(x) is the scrambling polynomial and d in x^d is the degree of d(x). By properly choosing d(x), scrambled sequences with balance of marks and spaces and adequate timing information, can be achieved. The bit stream of length j in polynomial version is described by $P(x) = C_{j-1}x^{j-1} + \ldots, + C_o$. If S(n) is a continuous bit stream and d(x) is the scrambling polynomial, and d(x) is the quotient obtained by dividing S(n) by d(x), the transmitted scrambled sequence is $tx(x) = Q_{d(x)}\left\{S(x).x^d\right\}$. Q is the quotient obtained

by dividing $S(x).x^d$ by $d(x)$. If there is an error pattern $e(x)$ during transmission, the decoded sequence, $u(x)$ will be,

$$\begin{aligned} u(x) &= Q_{x^d}\{rx(x).d(x)\} \\ &= Q_{x^d}\{tx(x) + e(x)\ d(x)\} \\ &= S(x) + Q_{x^d}\{e(x).d(x)\} \end{aligned}$$

Thus, in the absence of transmission errors, accurate recovery of the source bit stream is possible. Scrambling does not introduce any redundancy. Guided Scrambling [9] is another technique which takes care of the problems with scrambling and has the capability to produce sequences for transmission in a line coding milieu.

1.5 The PF mB(m+1)B Code

Unlike the mBnB codes, the mBIC and DmBIM codes cannot be represented by the FSMmodel as they do nothave a state diagram representation having a finite number of states. In these codes, the RDS is not constrained and hence is unbounded. Another code PF mB(m+1)B (Partially Flipped mBm+1B) code is an improvement over the above two codes. In this code, the code sequences are balanced and the RDS is bounded. The input binary sequence is grouped into blocks of m bits each. Each word is precoded into words m+1 bits long. The (m+1)th bit, is a '1' if there are a majority of 0's in the m bits and it is a '0' if there are majority of 1's inthe m bits. The m+1 bit word thus formed is coded into a codeword for transmission. The bits C_i, i = 1,2,..., m+1 are chosen as

$$C_i = \begin{cases} B_i & \text{if } D^{(n)}_{m+1} \times WRDS(n-1) \leq 0 \\ \bar{B}_i & \text{if } D^{(n)}_{m+1} \times WRDS(n-1) > 0 \end{cases} \qquad i = 1,2,\ldots,m$$

$$C_{m+1} = B_{m+1}$$

B_i's are the bits of the (m+1) bit word before coding.

WRDS(n-1) is the word end Running Digital Sum at the end of (n-1) words.

D_{m+1} is the disparity of the nth (m+1) word before coding.

After reception, if the majority rule, which was applied before coding,is satisifed, the received bits directly give m bit input sequences, if it is not satisfied, we have to complement the bits to yield m bit input sequences. The run length for this code is 2m+1. The RDS of the code alternates between $\frac{3m-1}{4}$ and $\frac{-3m-1}{4}$. The code is dc free. The decoding rule of PF mB(m+1)B code leads to an error extension effect.

In the above discussion we considered only binary codes such as mBnB, mBIC, DmBIM and PF mB(m+1)B, because in optical fiber systems, binary transmission is preferred over M-ary systems. Due to the signal dependent shot noise in such systems, decision levels would require to be nonuniformly spaced if multi level signals are transmitted. Also, decision and regeneration of multi level signals is more complicated than those for binary signals.

1.6 Introduction to Error Conrol Coding

When we model the communication channel as a Binary Symmetric Channel (BSC) or a Binary Erasure Channel (BEC), there

is a finite probability with which a bit can be erroneously received. The decoding failure can occur because of channel characteristics like low frequency cutoff, where the received bits appear as decaying amplitudes. Eventhough, normally the probability of erroneous decoding is low in fiber optic transmission, it is advantageous to use Error Control Coding to provide reliable communication especially between computers with fiber optic links.

Error Correcting Codes are divided into two broad categories, namely block codes and convolutional codes. The block codes can be linear or cyclic codes. Linear block codes are one in which linear combination of the codewords is also a codeword. In cyclic codes, a cyclic shift of the codeword is also a codeword. Systematic codes are on in which in every codeword, the information part is retained as it is The decoding is easier for such codes.

In the following, we consider block codes of both Error detecting and Error Connecting (EC) types. In this, we are mainly concerned with the problems of communication. Codes which are pertinent to this work and also applicable in areas like magnetic recording, have been garnered in a coherent manner. Some modified codes with EC capability are also introduced here.

1.7 Organization of the Thesis

The thesis is organised as follows :

In Chapter Two, discussions on Error detection and correction capabilities of the codes proposed by various authors are included.

Chapter Three contains discussion on modification of existing codes which we have suggested. They are efficient and emphasize techniques which are not used in the presently available dc constrained code design.

In Chapter Four, we make a comparison of the directly coded 8B10B code with the combined 8B10B code. This is followed by a discussion on the dc constrained code efficiency.

Chapter Five introduces the guided scrambling concept.

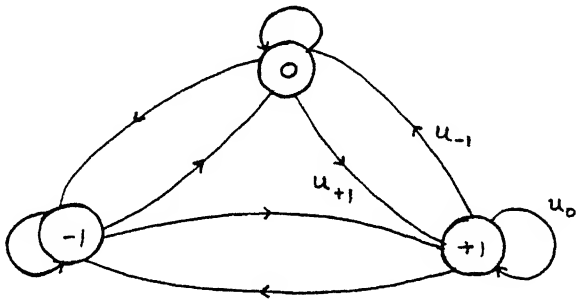
Finally, the conclusions are presented in Chapter Six.

Suggestions for further work are also included here.

CHAPTER - II

INTRODUCTION

In practice, the line coder is modelled as a Finite State Machine (FSM). The message sequence is split into blocks, say m bits each and each block is coded into n bits. Since the line code must satisfy the rules discussed previously, an alternative codeword of opposite disparity is provided to each condeword. The FSM of an encoder might look as shown below.



u_0 - Codeword with 0 disparity

u_{-1} - Codeword with -1 disparity

u_{+1} - Codeword with +1 disparity.

each state of the FSM is determined by one of the all possible values of the word end Running Digital Sum (WRDS) of the code sequence. Suppose one have three states given by 0, +1 and -1 as shown above. If the coder is in state -1 and a codeword u_0 is encoded, the coder remains in state -1. If a codword u_{+1} is encoded, the resulting state is 0 and so on. Thus, the incoming message sequence results in a transition to the same or another

state. The encoding rule ensures an alternating RDS between two opposite levels so that the resultant code sequence is dc free. More details on FSM and its use can be seen in [14]

The size of the FSM model and the analysis can be made simpler if there are fewer states and the simplest model is one in which there is a single state. If this lone state corresponds to a WRDS $\geq$ 0, the code has a constant dc If the state corresponds to WRDS = 0, the code is dc free If the code has constant dc, the level can be restored using dc recovery circuits. Balanced codes are more advantageous because they do not require the additional circuitry. Also, balanced codes provide powerful error detection.

The balanced codes proposed by Knuth [1] and Bose [2] serve the purpose in a telecommunication environment. These codes are of error detecting type only, but their encoding and decoding algorithms are simple. The work on balanced codes was first done by Knuth. We give some of the important results below of his work.

2.1 Efficient Balanced Codes

Let n be the number of message bits and P be the number of parity bits. The length of the code is (n+P). Let $M(m) = \left\lfloor \frac{m}{\frac{m}{2}} \right\rfloor$ be the total number of balanced words of length m and weight ($\frac{m}{2}$). To provide adequate number of balanced words, we should have $M(n+P) \geq 2^n$. Using Stirlings approximtion, the optimum value of n is chosen to be 2^P. Thus, in the communication environment we

have a code set with blanaced code words with increase in code rate $n+P/n = 1 + P/2^P \simeq 1$ if P is even and RDS $\leq \frac{n+P}{2}$ and WRDS = ±1 if P is odd Thus, if P is odd, we have to chose complementary pair of code words from which the transmitted codeword is to be chosen such that we confirm to the rules of alternate disparity word transmission.

PARALLEL SCHEME :

$v(w)$ = total numberof 1's in binary word w.

$v_k(w)$ = number of 1's in the first k bits of w.

$w^{(k)}$ = word w with first k bits complemented.

Thus, we have $v(w^{(k)}) = v(w) + k-2v_k(w)$.

If w has length n and let $\sigma_k(w)$ stand for $v(w^{(k)})$ the quantity $\sigma_k(w)$ changes by ±1 when k increases by 1. It describes a "random walk" from $\sigma_o(w) = v(w)$ to $\sigma_n(w) = n - v(w)$. Since $\left[\frac{n}{2} \right]$ lies in the closed interval v and n-v for all integer v, there always exists a k such that $\sigma_k(w) = \left[\frac{n}{2} \right]$. In other words every word w can be balanced in the above manner. If we encode k bits in a balanced word u of length P, and if n and P are not both odd, we can let w correspond to the balanced codeword $uw^{(k)}$.

For example, if wehave an 8 bit information word w, we can ind a k such that $w^{(k)}$ is balanced, the number of parity bits is rbitrarily chosen to be 5. Parallel decoding makes use of a ookup table. Thus we have a look up table with eight words each ord 5 bits long. The eight words correspond to k = 0,8 and ,2,3,4,5,6,7. Thus the look up tableis very small in size.

SERIAL SCHEME :

This is an improvement over the previous scheme in that any imbalance in n is compensated by a corresponding imbalance in $w^{(k)}$, for n = 8 and P = 3, the following look up table can be used.

v(w)	u	s	v(w)	u	s	v(w)	u	s
0	001	4	3	101	3	6	111	2
1	011	3	4	100	4	7	110	3
2	010	4	5	000	5	8	001	4

here $s = v\left[w^{(k)}\right]$.

he word $uw^{(k)}$ will be balanced if and only if $v\left[uw^{(k)}\right] = v(u) + v(w) = \left[\frac{11}{2}\right] = 5$. The code is defined by choosing smallest k such hat $\sigma_n(w) = s$.

To decode this scheme, first v(w) is determined from u, then ie smallest k is found out such that $\sigma_k(w) = v(w)$. Thus the bservation in the above schemes is that they are error detecting ly because of the minimum distance property, an error is tected if v(w) is not tallied by that implied by u and vice rsa. Also since the rate increase is not much significant they n be used for low speed transmission systems. The maximum run ngth is $\frac{n+P}{2}$.

An extension of the work done by Knuth is taken up by Bose [2 From his discussion, an important use of the balanced code is served. Even if, they do not provide ever correction they can

detect all unidirectional errors. The necessary properties for unidirectional errors detection is discussed later. We can model the channel in which only 1 $\longrightarrow$ 0 or 0 $\longrightarrow$ 1 transitions take place. If we are using one of such channels, we have two advantages, such codes provide the desired dc characteristics as well as detect all unidirectional errors. In the next section the method suggested by Bose is described in detail.

2.2 BALANCED CODES WITHOUT ERROR CORRECTION

Let k - number of information bits

r - number of check bits

ko - number of 0's in the information part

k1 - number of 1's in the information part

PARALLEL CODING SCHEME :

If we have 8 bit long information symbols, there can be words of weight 0,1,2, ... 8. There is an all zero word and an all one word. If we complement first 4 bits, we get 11110000 and 00001111 and the check symbol is 011 for both.

The check symbols for other information symbols are based on the number of 0's. If ko = 7,6,5 or 4, check symbol is ko in binary. If ko = 3,2, or 1, check symbol is ko-1 in binary. Thus, we have,

w(x)	check	word	modified	check
1	111	00000000	11110000	011
2	110	11111111	00001111	011
3	101			
4	100			
5	010			
6	001			
7	000			

Thus, in general,

(1) if ko = 0 mod 2^r, Complement 2^{r-1} first bits and append $2^{r-1}-1$ in binary as check.

(2) if $2^{r-1} \leq$ ko $\leq 2^r-1$, Append ko in binary as check

(3) if $1 \leq$ ko $\leq 2^{r-1}-1$, Append ko-1 in binary as check.

Thus we obtain balanced codes, but there is no error correction.

Example :

k = 11, r = 3, 7 out of 14 code is constructed.

Words of weight imod8 where i = 0,1,2,3 can be mapped into words of weight 4,5,,7 for words of weights J = 4,5,6,7 words can be formed of weight in the range J and 11-J. One interesting characteristic of this particular code is that the check symbol directly represents weight imod8 of the information symbol.

w(x)	0	1	2	3	4	5	6	7	8	9	10	11
w(F(x)	7	6	6	5	6	5	5	4	7	6	6	5
check	000	001	010	011	100	101	110	111	000	001	010	011

Decoding : If check symbol value is i, $0 \leq i \leq 7$, complement the first 0,1,2,...11 bits until the new word has weight imod8. Find

J such that $\sigma_J(x)$ = imod8 and complement the first J bits of x to get original information symbol where $0 \leq J \leq 11$.

The general construction method is as follows :

Let the number of information bits be $k = 2^r+J$ where $0 \leq J \leq 2^r-(r+2)$ for $r \geq 2$.

Convert all information words of weight i and 2^r+i to words of weight w for all i = 0,1,2,... J and then append r bit check symbol such that $w + w(cs) = \left[\frac{k+r}{2}\right]$. Similarly, words of weight j, j = J+1, J+2,..., 2^r-1 are mapped into words of weight w_1 where w_1 is in the range [J, K-J] and append r bit check symbol such that,

$$w_1 + w(cs) = \left[\frac{k+r}{2}\right].$$

The information symbols with weight other than $\left[\frac{K}{2}\right] \pm i$, i = 0,1,2, ..., $\left[\frac{r}{2}\right]$ can be mapped into words of weight $\left[\frac{k}{2}\right] + l$ and a check symbol of weight J is added such that

$$l + J = \left[\frac{r}{2}\right]$$

We have seen in the above discussion, various methods to design balanced codes. The simplicity of the codes lies in the fact that encoding and decoding tables are very small compared with the standard arrays for decoding EC linear block codes. These codes are only error detecting which is provided by the balance ofthe code words. Since there is very little redundancy introduced the increase in code rate is kept low.

In the next section, we will see the design of dc free codes which provide both error detection and correction.

2.3 DC FREE - COSET CODES

The motivation for the work on dc constrained codes for fiber optic link communiation, was the work done by Herro and Deng [3]. Interest in the above area has received much attention lately. There are some constraints for line coding (discussed previously), which have to be incorporated while designing codes. Thus error correcting codes which incorporate these features are of much interest because only error detecting codes which handle the pertinent features have been developed till data. The codes designed by Herro and Deng incorporate zero dc spectrum, limited run length properties. Line codes which have zero dc spectrum and limited run length have been designed previously but, they have only error detection capability.

Notations :

D = maximum running disparity after any bit position.

D^1 = running disparity at the end of a codeword.

Usually run length is denoted by (ℓ,L) where ℓ = minimum run length -1, L = maximum run length -1; usually in the digital transmission system of concern, ℓ = 0.

The dc free coset codes are denoted by (n,k,D) where

n = codeword length, k = information length, in bits.

2.3.1 DC Free Coset Codes with Error Detection only

An n-bit codeword v is given by

v = (o,u) +a1n

where u = (n-1) bit information word

a = 1 or 0

1n = n bit all 1 vector.

The coset code consists of linear code

$$T_1 = \{v_i / v_i = (o,u)\},\ a = o$$

and its coset

$$T_2 = \{v_i' / v_i' = v_i + 1n\} \quad \text{for } i = 0,1,\ldots,2^{n-1}-1.$$

where v_i' is the complement of v_i.

The construction of dc free coset code is based on "vector space partitioning". The linear space of 2^n vector is partitioned into 2^{n-1} disjoint subsets $\{A_o, A_1, \ldots. A_{2^{n-1}-1}\}$ where $A_i = \{v_i, v_i'\}$ for $i = 0,1,\ldots, 2^{n-1}-1$.

Let D_t' we the running disparity at the end of a codeword at time t If an (n-1) bit information vector is to be encoded at time t, a subset A_i is chosen, and D_{t-1}' is used to select a codeword in A_i.

DC FREE COSET CODES WITHOUT ERROR CORRECTING CAPABILITY :

Encoding : Suppose an (n-1) bit information vector n, is to be encoded at time t. Let D_{t-1}' be the running disparity at the end of a codeword at time t-1. The coding steps are as follows :

(1) $v \longleftarrow (o,u)$, $D_t' \leftarrow D_{t-1}'$; Let d_t be the disparity of k at time t.

(2) if $D_t'.d_t \leq 0$, $D_t' \leftarrow D_t'+d_t$ then go to 4; or else go to 3.

(3) $v = v+1n$, $D_t' \leftarrow D_t' - d_t$,

(4) encode next n-1 information bits.

Decoding :

Let $v = (v_1, v_2, \ldots, v_n)$ be the received version of v. The decoding algorithm is as given below .

(1) if $v_1 = 0$, $u = (v_2, v_3, \ldots, v_n)$ otherwise

$u = (v_2', v_3', \ldots, v_n')$

(2) decode next n bit received word.

The running disparity at the end of the any codeword is bounded by $|D'| \leq n$ The maximum running disparity at any given bit position is given by $|D| = n + \left[\frac{n}{2}\right]$ The worst case occurs when the disparity at the end of a codeword is 0, followed by an all 1 codeword, then followed by a codeword with $\left[\frac{n}{2}\right]$ 1's in its first $\left[\frac{n}{2}\right]$ positions. The maximum run length $L = 2n + \left[\frac{n}{2}\right] - 1$ and the worst case occurs when the disparity at the end of a codeword is n, followed by two all 0 codewords, then followed by a codeword with $\left[\frac{n}{2}\right]$ 0's in its first $\left[\frac{n}{2}\right]$ position.

2.3.2 DC Free Codes with Error Correcting Capability

In this section, the previous ideas are extended to provide an improvement in the capabillity ofthe code.

The code is defined by (n,k+J) where J is the number of 0's added to the MSB of the information word which is then converted into a codewords n bits long. This can correct t errors and simultaneously detect λ or fewer errors provided that $2t+\lambda+1 \leq d_b$, where d_b is the minimum distance of the code. Though, linear block codes can be designed for powerful error correction, they do not have good dc properties. The following code provdes a compromise between the two properties.

The dc free coset code is defined by

$$v = u(G_2G_1) + g.$$

where u is a k bit information vector, G_2 is a kx(k+J) matrix called the transfer matrix. The matrix G_2 transfers the k bit information vector into k bits of a (k+J) bit vector with the first J bits 0's and $g = \sum_{J=1}^{J} a_j g_j$, $a_j = 0$ or 1.

Thus, g is a linear combination of first J rows of (k+J)xn matrix G_1 which is the generator matrix. The code word g is used to control disparity. The first J rows of G_1 are chosen to satisfy supp $(g_i) \cap$ supp $(g_j) = \phi$ for $i \neq j$ and $g_1+g_2+ \ldots + g_J = 1n$ supp(g_i) is the set of coordinates at which the components of g_J are nonzero. Thus g_J controls disparity of coordinates supp (g_J) and $g_1+g_2+ \ldots + g_n = 1n$ guarantees that the disparity at all the coordinates of v is controlled.

G_2 is given by a kx(k+J) matrix of the form

$$\begin{bmatrix} 0 & 0 & \\ 0 & 0 & \\ 0 & 0 & I_k \\ \vdots & \vdots & \\ 0 & 0 & \end{bmatrix}$$

I_k is kxk identity matrix.

Thus, we have,

$$v = uG_2G_1 + g = (0_J,u)G, g = \sum_{j=1}^{J} a_j g_j$$

$$= (a_1a_2 \ldots a_Ju)G_1,$$

where 0_J stands for J zeros. Also $G_2G_2^T = I_k$.

The information vector can be retrieved from $u = (a_1a_2 \ldots a_Ju)G_2^T$.

Encoding :

A k bit information vector can be encoded at time t as follows :

(1) $v \leftarrow uG_2G_1$, $J \leftarrow 1$, $D'_t \leftarrow D'_{t-1}$

(2) if $D'_t.d_J \leq 0$, $D'_t \leftarrow D'_t+d_j$ then $j \leftarrow j+1$ and go to 4; or else go to 3.

(3) $v \leftarrow v+g_j$, $D'_t \leftarrow D'_t-d_j$, $j \leftarrow j+1$;

(4) if $j \leq J$, then go to 2; else encode next information block.

Decoding :

Let $\hat{v}$ be the received version of $v = (a_1a_2,\ldots, a_ju)G_1$

(1) Find $(\hat{a}_1,\hat{a}_2,\ldots, \hat{a}_j\hat{u})$ based on block code with generator matrix G_1;

(2) $\hat{u} = (\hat{a}_1, \hat{a}_2,\ldots, \hat{a}_j\hat{u})\ G_2^T$

(3) decode next received word.

Properties of the Code :

There will be an obvious improvement here in the dc propertiesof the code became the all zero and are words canalso be modified. Thus, we have, $|D'| \leq \max w_j$ where w_j is the Hamming weight of g_j. It is stated using analogous reasoning, that

$$|D| \leq w_{max} + \left[\frac{w_{max}}{2}\right] \text{ and } L \leq 2w_{max} + \left[\frac{w_{max}}{2}\right] - 1$$

where $w_{max} = \max w_j$

Also a bound on the number J is derived as

$$J \leq \left[\frac{n}{d_s} \right].$$

Further details of these codes were provided by the Popplewell and Reilly [8].

2.4 ERRROR CORRECTING (16,8) BALANCED CODE

In the balanced codes we have seen that, since the minimum distance is 2, we do not have an error correction. Now, an improvement over those codes is to increase the minimum distance and at the same time provide run length constraints. Work in this area has been done by Ferriera [4] and an improvement has been proposed by Balum [5]. In the following section, we provide some details of both the codes designed by them. Ferriera proposed a (16,8) d_{min} = 4 dc free block code. With run length maximum = 8 and maximum RDS = 5. This is particularly useful because memory elements with 16 or 8 address lines are readily available.

The 16 bit codeword is represented as follows :

$\overline{cw} = a_1a_2\ b_1b_2\ c_1c_2\ d_1d_2$ and let 2-tuples (n_1,n_2), n = a,b,c or d in $\overline{cw}$ contain only elements of A and the other two contain only elements of B where

$$A = \{ 00,\ 11 \} \qquad \text{and} \qquad B = \{ 01,\ 10 \}$$

Let $w_\alpha(\overline{cw})$ be the height of element α in $\overline{cw}$.

Let $w_{oo}(\overline{cw}) = w_{11}(\overline{cw})$ in order for $\overline{cw}$ to be dc free and also let $w_{01}(\overline{cw}) = w_{10}(\overline{cw})$ be even.

Since two of the four 2-tuples in $\overline{cw}$ must contain only elements from A and theother to only elements from B, there are $\binom{4}{2}$ = 6 different ways in which form 2-tuples can be chosen from A and B. Sin $w_{oo}(\overline{cw}) = w_{11}(\overline{cw})$, there are $\binom{4}{2}$ = 6 ways in which theese symbols can be chosen from A. Since $w_{01}(\overline{cw})$ and $w_{10}(\overline{cw})$ are always even, there are $\left\{\binom{4}{0}+\binom{4}{2}+\binom{4}{4}\right\}$ = 8 ways in which the form symbols of $\overline{cw}$ which are elements of B can be chosen from B. Thus, the total number of different codewords are 6.6.8=288 > 256 = 2^8. Thus a code with k = 8 can be formed.

From the set of 288 codwords, 256 are selected which are all balanced. The maximum run length can occur when a code ending with $C_2 d_1 d_2$ = 100000 is followed by a word starting with $a_1 a_2 b_1$ = 000001; hence ℓ = 9. The maximum value of RDS is 5 which occurs when $a_1 a_2 b_1$ = 000001.

By omitting 32 out of 288, words, the run length can be reduced to 8, hence (b,ℓ,c) = (0 7 5).

ENCODING AND DECODING :

The 256 different information words can be mapped onto the codwords by means of a 256x16 bit ROM. This completes encoding.

DECODING :

Since d_{min} = 4, a single bit error can be corrected. Decoding is done by implementing a 64Kx8 bit look up table for ROM's.

(1) Examine the four 2 tuples (x_1x_2) in the received word. Detect the 2 tuple in error since it does not contain only elements of A and B.

(2) Determine the set A or B from which the elements of the 2 tuple in error were chosen. If there are two other 2 tuples with only elements of B and one other 2-tuple with only elements of A, with 2-tuple in error should contain only elements of A.

(3) Locate the symbol in error, since it is the symbol in the 2-tuple which is not an element of the set A or B as determined in (2).

(4) If the symbol in error should be an element of A, invert one of the two bits of the symbol such that dc balance of the received word is restored.

(5) If the symbol in error should be an element of B, invert one of the two symbol bits such that $w_{01}(\overline{cw})$ and $w_{10}(\overline{cw})$ is even.

(6) Map the corrected received word onto the corresponding information word by means of 256x8 bit PLA with 16 input lines

We see that the increase in code rate is 2 and a corresponding increase in Bandwidth of similar magnitude. The run length is 8 which is fairly large. Another (16,9,6,5,4) code is proposed by Blaum [5] in which the increase in code rate is reduced from 2 to 1.83. This is also a single error correcting code since $d_{min}=4$. In the following section, the code proposed by Blaum is discussed in detail.

2.5 DC FREE ERROR CORRECTING BALANCED CODE

The code is defined by the parameters (2n,k,l,c,d). 2n is the length of the codeword, each of weight n. In other words, the codewords are balanced words. k is the information length, l is the run length; c is the RDS, and d is d_{min}. Concatenation of an error correcting code and a dc free code can be of great use in a fibre optic channel where the codeword takes care of some of the properties pertinent to such a channel. Associate symbol 0 with disparity f(0) = -1 and '1' with f(1) = +1.

Construction of (16,9,6,5,4) code :

Consider each codeword as formed by eight 2-tuples. Fill each 2-tuple with 10 or 01 in such a way that the number of 10's is even. This is referred to as type I code, and there are 128 such codewords.

Next consider each codeword as formed by four blocks of four bits each. Choose any two blocks and place any two 4-tuples of weight 1 Take the first of the remaining two 4-tuples and place a 4-tuple of weight three. Take the first block of weight 1. See how many time it has to be rotated to the right to make it complement of 4-tuple of weight three. Place a 4-tuple of weight 3 which is the complement of the 4-tuple-1 obtained by rotating the second 4-tuple-1 that many times.

There are $\binom{4}{2} 4^3 = 384$ such codewords. These are type II codewords. Thus the total number of codewords are 128+384 = 512.

Thus the message bits are 9. The maximum run length is 6. The maximum RDS is |5|.

ENCODING AND DECODING :

Let the 9 bit information vector be $a_1 a_2, \ldots, a_9$.

If $a_1 a_2 = 00$, it will be encoded as type I.

$a_{10} = \sum_{i=3}^{9} a_i \bmod 2$ and the codeword is

$c = a_3 \bar{a}_3 a_4 \bar{a}_4 \ldots\ldots a_{10} \bar{a}_{10}$.

If $a_1 a_2 \neq 00$, it is coded by type II.

We have six possible choices for $a_1 a_2 a_3$

The two blocks in which we place 4 tuples of weight 1 are chosen as shown below :

$a_1 a_2 a_3$	
0 1 0	AB
0 1 1	AC
1 0 0	AD
1 0 1	BC
1 1 0	BD
1 1 1	CD

$a_4 a_5$ will determine 4-tuples in first block and $a_6 a_7$ will determine 4-tuples in second block.

00	$\longrightarrow$	1000
01	$\longrightarrow$	0100
10	$\longrightarrow$	0010
11	$\longrightarrow$	0001

$a_8 a_9$ can be viewed as binary representation of interger J where $0 \leq J \leq 3$. Consider the complement of first 4-tuples and rotate J times to the right. This gives first 4-tuple of weight 3. The same is done to second 4-tuple of weight 1 to get second 4-tuple-3.

Let $\hat{c}(\vec{a})$ be the received word. Assume that no more than one error has occurred. The first step is to determine whether the word is of type I or type II.

If the blocks A, B, C, D in $\hat{c}(\vec{a})$ are having weight 1 or 3 in majority, it is of type II, or it is of type I.

Assume $c(\vec{a})$ is of type I, if an error occurs, one of the 2-tuple is either 00 or 11. Consider the remaining 2-tuple and count how many times 10 appears. If this is even, the error 2-tuple is 01; if odd the error 2-tuple is 10.

Assume $C(\vec{a})$ is of type II.
Since there is one error, one block out of 4 will have even weight.

Let the four bit blocks H_1 and H_2 in $C(\vec{a})$ have the same weight. Let G_1 and G_2 be the remaining two blocks. Let H_1 precede H_2 and G_1 precede G_2.

Let $\rho_J(K)$ be the rotation of block K, J times to the right. We obtain $\vec{a}$ from $\hat{C}(\vec{a})$ by applying following algorithm :

(1) determine J such that $\rho_J(\bar{H}_1) = G_1$ or $\rho_J(\bar{H}_2) = G_2$, $0 \leq J \leq 3$.
(2) In $\hat{C}(\vec{a})$ replace G_1 by $\rho_J(\bar{H}_1)$ and G_2 by $\rho_J(\bar{H}_2)$. The vector obtained is the correct codeword $C(\vec{a})$.
(3) Obtain $\vec{a}$ from $C(\vec{a})$.

In this chapter we have seen the design of balanced codes followed by that of single error connecting balanced codes. The EC balanced codes are very useful because the code length is 16

and hence, memories which are usually organised in bytes can be efficiently used in size for lookup tables. Also, the EC balanced codes are more efficient than the EC dc free coset codes [3], because the increase in code rate is lesser in the case of balanced codes.

In the next chapter, we see a different approach for designing the dc constrained codes and we propose the codes with error correction followed by codes with both error detection and error correction. These codes are efficient.

CHAPTER - 3

MODIFIED DC FREE CODES

As already mentioned, line coding has been defined as a technique where the code incorporates features like (1) adequate timing information (2) disparity control (3) zero power spectral density at dc (4) balance of 1's and 0's in a codeword, etc. Line coding does not incorporate error correction, which may be desirable for fibre optic channels under consideration. Using the features of scrambling and linear Error Correcting Codes, we introduce a coding scheme which incorporates line code features.

3.1 DC CONSTRAINED CODES - WITH ERROR DETECTION ONLY

We use a linear (7,4) block code which has a minimum distance 3.

The parity check matrix $H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$

The generation matrix $G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$

The codeword table is given below :

Message	Codeword	Disparity
0000	0000000	-7
0001	1010001	-1
0010	1110010	+1
0011	0100011	-1
0100	1100100	-1
0101	0110101	+1
0110	0010110	-1
0111	1000111	+1
1000	0111000	-1
1001	1101001	+1

1010	1001010	-1
1011	0011011	+1
1100	1011100	+1
1101	0001101	-1
1110	0101110	+1
1111	1111111	+7

Out of the sixteen codewods, fourteen can be divided two blocks of disparities +1 and -1 respectively. Also the blocks are self complementary Thus one part of the enco process is achieved. The all zero and all one codewords are complementary, but not included, they cause a large run len Thus these two are to be modified.

The general technique of generating a self complemen code [13] is discussed bleow.

Let the parity check matrix

$$H = \begin{bmatrix} P_o \\ P_1 \\ \vdots \\ P_{r-1} \end{bmatrix} \quad \text{where } P_i = \begin{bmatrix} h_{i,o}, & h_{i,1} & \cdots & h_{i,n-1} \end{bmatrix}$$

If $\sum_{J=o}^{n-1} h_{i,J} = 0$, where Σ implies a mod2 addition.

Then it is called even weight row code.

Let a codeword $w = \{d_o, d_1, \ldots, d_{k-1}, P_o, P_1, \ldots, P_{r-1}\}$

Let $H = \{H' I\}$ $\quad H'_{rxr}$ is the information part

I_{rxr} is parity check part.

each parity bit can be obtained as

$$P_i = \{d_o, d_1, \ldots, d_{k-1}\} \left[W'_i\right]^T \qquad 0 \le i \le r-1$$

Let the component of $w = \bar{w} = \left[\bar{d}_o, \bar{d}_1, \ldots, \bar{d}_{k-1}, \bar{c}_o, \ldots, \bar{c}_{r-1}\right.$

The following holds for odd weight of every row vector h_i'

$$\left[\bar{d}_o, \bar{d}_1, \ldots, \bar{d}_{k-1}\right] \left[h_i'\right]^T = \bar{P}_i .$$

This implies, $\bar{D}\left[H'\right]^T = \bar{P}$

where $\bar{D} = (\bar{d}_o, \bar{d}_1, \ldots, \bar{d}_{k-1})$ and hence the code is complementary.

The all zero codeword is unavoidable because this l linear block code. We would like to convert all-one and all- codewords into other words, which suit our requirements. For purpose, we can invoke the technique of scrambling by using scrambling polynomial $d(x) = x^2+1$ which provides the hig transition density.

Thus, wherever all zero or all one words occur, the significant bit is complemented and the modified word scrambled. The process is shown below.
All zero word 0000000 becomes 1000000 which in polynomial for x^6. Thus the scrambled word is

$$S(n) = \frac{x^6 . x^d}{d(x)} = \frac{x^6 . x^2}{x^2+1} .$$

The quotient of division is $x^6+x^4+x^2+1$.

All one word 1111111 becomes 0111111 which in polyno form is $x^5+x^4+x^3+x^2+x+1$. Thus the scrambled word is

$$S(x) : \frac{(x^5+x^4+x^3+x^2+x+1).x^2}{x^2+1} .$$

The quotient of division is x^5+x^4+x+1 which is 0110011.

There are in all $\binom{7}{4} = \binom{7}{3} = 35$ words each of weight 4 and weight 3. But an exhaustive search shows that there is no other 7 bit word of weight 3 which is at a distance of at least 2 from the seven bit codewords of weight 4. Similarly, there is no other 7 bit word of weight 4, which is at distance of at least 2 from the seven 7 bit word of weight three.

Hence we choose the scrambled versions of all zero and all one words viz. 1010101 and 0110011 for transmission. Thus, we have 8words of disparity -1 and eight words of disparity +1. The next step is to develop and encoding table with two parts of opposite disparities. For this purpose, we add to the most significant bit of each codeword, a '0' or a '1'. The reason why we have done this is that we would like to constrain the word end RDS among the values 0, ±2. If this is not done and when there is need to, transmit a few -1 or +1 disparity words in a sequence, the RDS may go without bound. By doing the above modification, we have a choice to transmit a word from 0 or ±2 disparity words and hence the RDS also varies from -2 and +2 rendering the code dc free. The modified code table tooks as shown below.

Message	Disparity	Codeword	Disparity	Codeword
0000	0	01010101	+2	11010101
0001	-2	01010001	0	11010001
0010	0	01110010	+2	11110010
0011	-2	00100011	0	10100011
0100	-2	01100100	0	11100100
0101	0	00110101	+2	10110101
0110	-2	00100110	0	10010110
0111	0	01000111	+2	11000111
1001	0	01101001	+2	11101001
1010	-2	01001010	0	11001010
1011	0	00011011	+2	10011011
1100	0	01011100	+2	11011100
1101	-2	00001101	0	10001101
1110	-2	00101110	+2	10101110
1111	0	00110011	+2	10110011

The code characteristics are $|RDS|_{max} = 4$,

Maximum Run length = 6

Increase in code rate = 2

The encoder and decoder are as shown in Fig. (3a). The above code does not provide any error construction. It can only detect errors. The complexity of the system is fairly high.

It can be observed in the encoding table that each message word has a balanced word. An improvement from the above code would be to transmit only balanced words. In which case in the Finite State Machine representation of the code there is only one state and the code is absolutely dc free because the state represents zero disparity.

The codewords are balanced as described below :

whenever the disparity is -1, a '1' is added to MSB

whenever the disparity is +1, a '0' is added to MSB

The scrambled sequences are made 1010101 and 0110011 for all zero and all one message words, respectively. Upon reception, the MSB is discarded and rest of the word is decoded.

The encoder and decoder structure are as shown in Fig. 3(a).

Now, we describe a scheme for generating the balanced words.

For the (7,4) code, the parity check was given by

$$H = \begin{bmatrix} 1001011 \\ 0101110 \\ 0010111 \end{bmatrix}$$

Maintaining the characteristics of a self complementary code we add another row and column to the H matrix. The new parity check matrix is given by

$$H' = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

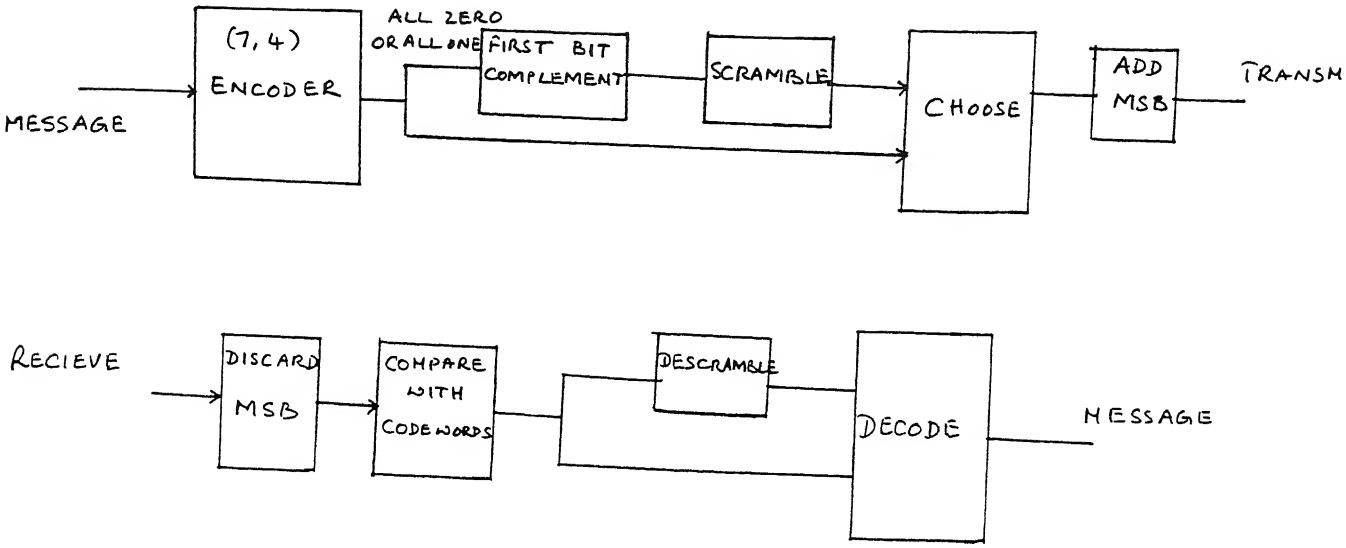
▭ represents the parity check matrix for (7,4) code.

Thus the generator matrix is given by

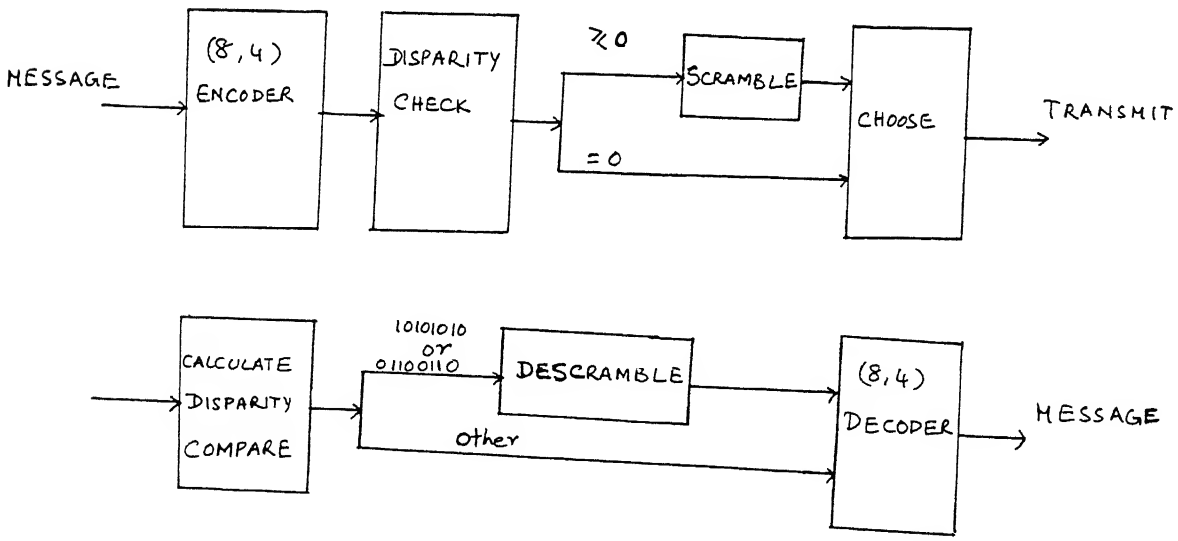
$$G = \begin{bmatrix} 11101000 \\ 10110100 \\ 01110010 \\ 11010001 \end{bmatrix}$$

The minimum distance of this code increases by 1 and hence it is 4

The code table is given below :


(7, 4)
ENCODER
MESSAGE
ALL ZERO
OR ALL ONE
FIRST BIT
COMPLEMENT
SCRAMBLE
CHOOSE
ADD
MSB
TRANSM
RECIEVE
DISCARD
MSB
COMPARE
WITH
CODEWORDS
DESCRAMBLE
DECODE
MESSAGE

3 (a)


MESSAGE
(8, 4)
ENCODER
DISPARITY
CHECK
≠0
=0
SCRAMBLE
CHOOSE
TRANSMIT
CALCULATE
DISPARITY
COMPARE
10101010
or
01100110
DESCRAMBLE
Other
(8, 4)
DECODER
MESSAGE

3(b)

Codeword message	Disparity
00000000	-8
11010001	0
01110010	↑
10100011	
01100101	
11000110	
00010111	
11101000	
00111001	
01001011	
01011100	
10001101	
00101110	0
11111111	+8

Using the same scrambling polynomial $d(x) = x^2+1$ all-zero and all-one words are transformed into 10101010 and 01100110 respectively. The encoder and decoder structures are shown in Fig. 3(b).

There is an improvement in the code characteristics

The maximum Run length = 6

$|\ RDS\ |_{max.}$ = 3

The minimum distance d_{min} = 2

3.2 ERROR CORRECTING DC CONSTRAINED CODES

It has been observed that if all possible constant weight words are used in designing a code, the minimum distance is 2 and hence error correction is not provided. If we want to increase the minimum distance, we have to omit some of the words.

We design a (6,2) d_{min} = 4 constant weight balanced code as described below.

We have $\binom{6}{3}$ = 20 weight 3, length 6 words.

These 20 words can be divided into six classes. Two classes are

of size four and four classes are of size three. Inside each class, the minimum distance is at least four.

Thus, we pick up a class of size four and design a (6,2) systematic code.

We start with an ordinary (6,2) linear block code. Let the generation matrix

$$G = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Thus the code set is

$$C = \begin{array}{c} \begin{matrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

To eliminate the all zero word, we make

$$c_3 = \bar{c}_1,\ c_4 = \bar{c}_2,\ c_5 = c_1 + c_2,\ c_6 = \bar{c}_5$$

Thus we have a (6,2) balance, weight 3, $d_{min} = 4$ code. The code set is

$$C = \begin{matrix} 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{matrix}$$

The encoding table has 2^{n-k} rows and 2^k columns where k is the message length and n is the code length. As we run down the table we encounterthe erroneous versions of the actual codewords which are at the top of the table The erroneous versions can provide characteristics likelimited Run length and balance. Hence, these words are suitable for transmission and this is what is done above, a specific coset is chosen and the error control capability is tested.

For the chosen generator matrix, the parity check matrix is

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Thus the syndrome bits are given by

$$s_1 = c_1+c_3, \quad s_2 = c_2+c_4, \quad s_3 = c_1+c_2+c_5, \quad s_4 = c_1+c_2+c_6$$

and $s = (s_1 s_2 s_3 s_4)^T$ is the syndrome.

Since we have modified the c_3, c_4, c_5 and c_6 bits, the syndrome equations also change. Hence,

$$s_1' = c_1+c_3+1, \quad s_2' = c_2+c_4+1, \quad s_3' = c_1+c_2+c_5, \quad s_4' = c_1+c_2+c_6+1$$

The addition is mod-2.

and the new syndrome $s' = (s_1' s_2' s_3' s_4')^T$

Since $d_{min} = 4$, single error correction is provided. The code is dc free and the maximum Run length is 3 The maximum value of RDS is 2 and hence the DSV is 4.

The decoding can be done as shown below

s_1'	s_2'	s_3'	s_4'	bit in error
0	0	0	0	none
1	0	1	1	c_1
0	1	1	1	c_2
1	0	0	0	c_3
0	1	0	0	c_4
0	0	1	0	c_5
0	0	0	1	c_6
0	0	1	1	Multiple errors
0	1	0	1	
0	1	1	0	
1	0	0	1	
1	0	1	0	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

3.2.1 Conditions for Unidirectional Error Correction

One of the advantages of constant weight code is it can detect all unidirectional errors [ ] as the theorems explain below.

Theorem :

Let x and y be two codewords in c. Let c_j be set of codewords in c of constant weight j.

If $x \in C_{i+k}$ and $y \in C_i$, then,

(a) if k=0, $N(x,y) \geq t+1$, $N(y,x) \geq t+1$

(b) if $1 \leq k \leq 2t$, $N(x,y) \geq t + \left[\frac{k+1}{2} \right]$, $N(y,x) = t - \left[\frac{k-1}{2} \right]$

(c) of $1 \geq 2t+1$, $N(x,y) \geq 2t+1$

N(x,y) is the number of $1 \longrightarrow 0$ cross overs from x to y.

Proof :

(a) if k = 0, x and y belong to C_i and the cross overs occur pairwise and N(x,y) = N(y,x)

$$d(x,y) = N(x,y) + N(y,x) \geq 2t+1$$

$$\therefore \quad N(x,y) = N(y,x) \geq t+1$$

(b) if k > 0, x and y are in different sets C_{1+k} and C_i. The number of 1's in x are exactly k more than that in y. There must be at least k cross overs from x to y.

i.e. $N(x,y) \geq k$. The remaining (n-k) bits of x and (n-k) bits of y have same number of 1's. Thus cross overs in the remaining part occur pairwise.

$$N(x,y) = k + N(y,x)$$

$$D(x,y) = N(x,y) + N(y,x) \geq 2t+1$$

$$N(y,x) \geq t - \left[\frac{k-1}{2}\right] \text{ and } N(x,y) \geq t + \left[\frac{k+1}{2}\right].$$

(c) $N(x,y) = k + N(y,x) \geq 2t+1$

The necessary and sufficient condition for a code to detect all unidirectional errors is

$$N(x,y) \geq t+1, \quad N(y,x) \geq t+1 \ \forall \ x,y \in C.$$

For a balanced code which can correct t errors, this rule is satisfied.

3.2.2 Properties of Product Codes

Coset codes are also called Run length limited (RLL) codes. To improve the correction capability of the code, product code approach has been devised.

If C_1 and C_2 are two linear codes with parameters (n_1,k_1) and (n_2,k_2), then the product code C has parameters (n_1n_2,k_1k_2) and minimum distance d_1d_2 where d_1 and d_2 are d_{min} of C_1 and C_2 k_2 blocks of length k_1 are encoded with C_1. From the k_2xn_1 array thus formed, each column is encoded into codeword in C_2. Thus we have n_2xn_1 array which is the codeword of C. Each row in a codeword in C_1 and each column is a codeword in C_2. The resulting array is transmitted column or row wise.

Property 1 :

The product code can correct errors upto $\frac{d_1d_2-1}{2}$ and also any error burst of length $b = \max(n_1t_2, n_2t_1)$ where $t_2 = \frac{d_2-1}{2}$, $t_1 = \frac{d_1-1}{2}$.

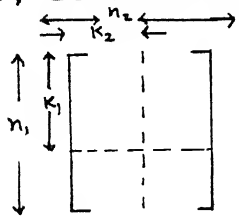
Property 2 :

If b_1 and b_2 are the burst error correcting capabilities of C_1 and C_2, then product code can correct at least all error bursts of length $b = \max(n_1 b_2, n_2 b_1)$. Usually C_1 is chosen to be a Run Length Limited (RLL) code and C_2 is any Error Correcting code and transmission is done row wise Such codes can be systematic or non-systematic and they introduce considerable redundancy.

We introduce a new method of designing such array codes. The codes here are systematic.

3.2.3 Product Code approach to Construct DC Constrained Codes

We choose C_2 to be a systematic RLL code and C_1 need not be an error correcting code. The systematic RLL code design has been discussed previously. The code array is as shown below. The information bits are arranged in the first k_1 rows and k_2 columns. The k_1 rows are encoded as codewords in C_2. The resulting n_2 columns are encoded as codewords in C_1. If C_1 and C_2 are both linear, it does not matter which is encoded first.



Since redundancy is to be kept as low as possible, for the column codes we choose two parity bits, one complementary to other, to provide balance and the parity bits indicate even and odd parity respectively. Thus, the row codes are RLL codes and

The code array looks as shown below :

$\leftarrow k_2 \rightarrow$

$$\begin{matrix} \uparrow \\ k_1 \\ \downarrow \end{matrix} \left[\begin{array}{ccc|c} 0 & 0 & & 0 \\ 1 & 0 & & 1 \\ 1 & 1 & & 0 \\ \hline 0 & 1 & & 1 \\ 1 & 0 & & 0 \end{array}\right]$$

Any two codewords must differ in at least one position (at least one information bit). Let this be in the ith row of the two codewords, where, $1 \leq i \leq k_1$ The minimum distance for this array code is d_r, where d_r is the minimum distance of the row code.

The difference between these codes and product codes are:

(1) The bottom n_1-k_1 rows need not be codewords in C_2.

(2) The codewords that result from encoding columns first and rows second and vice versa, need not be same.

(3) Column is a fixed code, whereas row code depends on the error control capability required.

(4) The transmission is only row wise because rows are RLL.

Now we discuss the decoding procedure of the above mentioned code.

Let the code under consideration be a single ECC "

(a) Check the first k_1 rows for errors. Let number of rows in error be r_1

(b) Check all the n_2 columns for errors. Let the numberof columns in error be r_2.

(c) If $r_1 = 0$ and $r_2 = 0$ or 1, there is no error in the information part.

(d) If $r_1 = r_2 = 1$, then there is an error in the intersection of row and column and can be corrected by complementing that bit.

(e) If r_1 and r_2 are greater than 1, there are multiple errors and they cannot be decoded (corrected) and hence error is detected.

Multiple Error Correction :

(a) Check first k_1 rows for errors. Let the number of rows in error be r_1.

(b) Check all columns for errors. Let number of columns in error be r_2.

(c) If $r_1 > t$ and $r_2 > t$, there are more than t errors and hence they cannot be corrected. (C_2 is assumed to be t error correcting code.)

(d) If $r_1 \leq t$ and $r_2 \leq t$ errors are corrected as follows. Consider $r_1 r_2$ positions at intersection of r_1 rows and r_2 columns. Correct the errors by complementing the bits in these positions.

This code can correct $k_1 t$ errors. If b_2 is the burst error correcting capability of C_2, then it can correct $k_1 b_2$ burst of errors.

The decoding of the array code can be more easily understood by the following example. Let the code matrix formed be

1	0	0	1	1	0
0	1	0	0	1	
1	1	1	0	0	
0	0	1	1	1	
1	1	0	0	0	

Let, r_1 and r_2 be two each after decoding; $r_1 = r_2 = 2$. In such a case, the box shown, represents the region in which the errors lie. Let us assume a single error correcting code. If there are more than one error in the two rows under consideration, they cannot be corrected. In such a case one arbitrary bit from each row in thebox is complemented (both made either 0 or 1) and decoding rule is applied, similarly other bits are alsotested for errror correction. Correction if possible is made, otherwise multiple errors are detected.

The row codes are to be chosen of even length and with balance of 1's and 0's errors in the columns are found by comparing parity with that indicated by the parity bit. The bottom (n_1-k_1) rows are not error correcting codewords in C_2. Hence, decoding of these rows is not done.

The decoding algorithm is simple with the added advantage that the code is systematic. The code is dc free because it is balanced The increase in code rate is n_1n_2/k_1k_2. The maximum run length is n_2 and hence maximum RDS is n_2-1.

In this section, we have seen the design of balanced codes, both error correcting and only error detecting, types. Proceeding

in this tenor, we now give a technique for selecting the code length (n) and message length (k) for such dc constrained codes.

Let C_i be the set whose elements are codewords each with weight i For any, Error Correcting Linear Block Code, if, C_o and C_n exist, $C_1, C_2, \ldots, C_{2t}$ and $C_{n-1}, C_{n-2}, \ldots, C_{n-2t}$ do not exist. This is because, they do not satisfy the minimum distance rule. Thus, if we make $2t+1 \leq n-2t-1$, which implies, $C_{2t+1} \leq C_{n-2t-1}$, there exist C_o, C_{2t+1} and C_n only. So, if $2t+1 \leq n-2t-1$, $n \geq 4t+2$ and $t \leq \frac{n-2}{4}$. The codewords are C_o, C_n and elements of $C_{n/2}$. The codewords of $C_{n/2}$ are balanced and only unbalanced words are the all-zero and all-one words. The minimum distance for such a code is $d_{min} \geq n/2$. For any linear block code, $d_{min} \leq n-k+1$

$$\therefore \frac{n}{2} \leq n-k+1 \quad \text{and} \quad k \leq \frac{n}{2} + 1$$

$$\therefore n \geq 4t+2, \quad k \leq \frac{n}{2} + 1.$$

The (6,2) balanced code designed previously satisfies this rule. There, all zero word is converted to weight-3 word using suitable transformation and all one word does not exist. We assume n to be even, all-one word will exist if there are odd number of 1's in each column of the generator matrix.

CHAPTER – 4

COMPARISON OF DIRECT AND COMBINED 8B10B CODE

4.1 DESIGN OF A COMBINED 8B10B CODE USING 3B4B AND 5B6B CODES

A new type of dc balanced, partitioned 8B10B transmission code has been proposed by Franaszek [6]. The beauty of this code lies in the fact tht the table look up is very small in size and the implementation is rather easy We discuss the important features of this code in the following.

Modern communication networks transmit information in the form of bytes with a defined field structure for address, information and error control The number of bits in each of these is found useful and it can be implemented more easily by suitable combining 3B/4B and 5B/6B codes.

The complexity of a code is defined by the number of information bits that must be examined by a coder in choosing a codeword. The various coding alternatives for the above mentioned code are as follows :

(a) The block code has parameters d = 0, k = 4, and v = 6 with rate 4/5, where

d = minimum run length – 1

k = maximum run length – 1

v = digital error variation.

Error propagation is limited to five bits

(b) A standard block code with no look ahead, with parameters

d = 0, k = 3, v = 5, s = 8. w = 10 and m = 1

s = number of information bits

w = number of code bits

m = number of s bit groups that must be considered while choosing a codeword

Error propagation is in general 8 bits

(c) A code with length 5 with k = 4, v = 5, s = 4 and m = 2 error propagation is five bits. However, there is little redundancy available here for suitable mapping of character.

Code Design :

Each in coming byte is partitioned into two sub-blocks. The first 5 bits are encoded into 6 bits and the next 3 bits are encoded into 4 bits. Thus the combined word has 10 bits. For the sub-blocks, the permitted disparity is 0, +2, -2. The coding rules require that the polarity of nonzero disparity blocks alternates No distinction is made between 6B and 4B sub-blocks. Nonzero disparity code points are arranged in complementary pairs to a single source data point. The encoding functions generate one of them, if it violates the alternating polarity rule, the complete sub-block is inverted. Determination of disparity and polarity in the 6B encoder is followed by corresponding operations on the 4B encoder, then the running disparity parameter is passed along for encoding of next byte.

The 3B4B and 5B6B encoding tables are as shown.

Table 3 5B/6B Encoding.

Name	*ABCDE K*	*Classifications*		*D-1*	*abcdei*	*D0*	*abcdei*
		Bit encoding	*Disparity*				*Alternate*
D.0	00000 0	L04	L22'•L31'•E'	+	011000	–	100111
D.1	10000 0	L13•E'	L22'•L31'•E'	+	100010	–	011101
D.2	01000 0	L13•E'	L22'•L31'•E'	+	010010	–	101101
D.3	11000 0	L22•E'		x	110001	0	
D.4	00100 0	L13•E'	L22'•L31'•E'	+	001010	–	110101
D.5	10100 0	L22•E'		x	101001	0	
D.6	01100 0	L22•E'		x	011001	0	
D.7	11100 0		L31•D'•E'	–	111000	0	000111
D.8	00010 0	L13•E'	L22'•L31'•E'	+	000110	–	111001
D.9	10010 0	L22•E'		x	100101	0	
D.10	01010 0	L22•E'		x	010101	0	
D.11	11010 0			x	110100	0	
D.12	00110 0	L22•I'		x	001101	0	
D.13	10110 0			x	101100	0	
D.14	01110 0			x	011100	0	
D.15	11110 0	L40	L22'•L31'•E'	+	101000	–	010111
D.16	00001 0	L04, L04•I	L22'•L13'•I	–	011011	+	100100
D.17	10001 0	L13•D'•I		x	100011	0	
D.18	01001 0	L13•D'•I		x	010011	0	
D.19	11001 0			x	110010	0	
D.20	00101 0	L13•D'•I		x	001011	0	
D.21	10101 0			x	101010	0	
D.22	01101 0			x	011010	0	
D/K.23	11101 x		L22'•L13'•I	–	111010	+	000101
D.24	00011 0	L13•D•E	L13•D•E	+	001100	–	110011
D.25	10011 0			x	100110	0	
D.26	01011 0			x	010110	0	
D/K.27	11011 x		L22'•L13'•I	–	110110	+	001001
D.28	00111 0			x	001110	0	
K.28	00111 1	L22•K	K	–	001111	+	110000
D/K.29	10111 x		L22'•L13'•I	–	101110	+	010001
D/K.30	01111 x		L22'•L13'•I	–	011110	+	100001
D.31	11111 0	L40, L40•I	L22'•L13'•I	–	101011	+	010100

Table 4 3B/4B Encoding.

Name	*FGH K*	*Classifications*		*D-1*	*fghj*	*D0*	*fghj*
		Bit encoding	*Disparity*				*Alternate*
D.K.x.0[a]	000 x	F'•G'•H'	F'•G'	+	0100	–	1011
D.x.1	100 0	(F≠G)•H'		x	1001	0	
D.x.2	010 0	(F≠G)•H'		x	0101	0	
D.K.x.3[a]	110 x		F•G	–	1100	0	0011
D.K.x.4[a]	001 x		F'•G'	+	0010	–	1101
D.x.5	101 0			x	1010	0	
D.x.6	011 0			x	0110	0	
D.x.P7	111 0		F•G, F•G•H	–	1110	+	0001
D.K.x.A7[b]	111 x	F•G•H•(S+K)	F•G, F•G•H	–	0111	+	1000

Evaluation of 8B10B code :

The maximum DSV for this code at arbitrary points is 6. The maximum DSV between sub-block boundaries is 2. All the 6B and 4B sub-blocks and the complete 10 bit characters have a disparity of either 0 or ±2. Each valid character in the LOB alphabet either has five 1's and five 0's or 6 1's and 4 0's or 6 0's and 4 1's. The maximum run length of this code sequences is 5.

Error detection :

One method for error detection is checking the disparity of the received word. Nonzero disparity blocks must have alternate polarity Some conditions as shown below can be attributed to errors.

$$a = b = c = d$$

$$f = g = h = j$$

$$e = i = f = g = h$$

where abcdeifghj is the codeword.

The simplest error patterns which may escape detection by the code are a single erroneous 1 complemented by a single erroneous 0. Such complementary errors when confined to a single word, may simply change it into another valid codeword. Thus it is possible that a complementary pair of digit errors can change disparity of the sub-blocks in conformance with alternating polarity rule, such that errors are not detectable by the code.

4.2 COMPARISON OF DIRECT 8B10B CODE WITH COMBINED 8B10B CODE

A close observation of the continued 8B10B code designed by Franaszek [6] shows that they had used 128 balanced and 128, ±2 disparity words of length 10 bits. In the following, we present an alternative 8B/10B code by direct transformation which makes use of 226 balanced words. For simplificationin forming the code table, we divide the 10 bit block into 4 bits and 6 bits sub-blocks. Codewords which have a repetitive pattern have been left out because they induce a pattern dependent Jitter. The codeword is abcdefghij.

We have 118 balanced words from set I, 52 balanced words from set II and 56 balanced words from set III.

Set I			Set II			Set III		
No.of code-words	abcd	efghij	No.of code-words	abcd	efghij	No.of code-words	abcd	efghij
20	0011	101010	13	1011	001100	14	0100	110011
19	0101	110001	13	1101	010100	14	1000	101011
20	0110	100011	13	1110	001001	14	0010	110110
20	1001	110010	13	0111	010001	14	0001	101110
19	1010	100110			001010			110101
		110100			011000			100111
		100101			101000			010111
		010110			100100			011011
		001011			010010			101101
		101001			000110			111001
		011001			000101			011101
		011010			100010			011110
		001110			100001			111100
		101100						
		001101						
		010101						
		011100						
		011100						
		111000						
		000111						

This code has a minimum distance 2.

The codewords of disparity -2 are

No.of codewords

13	1010	
13	1001	Combine 6 bit set of II.
4	1100	

Total = 30

The codewords of disparity +2 are the complement of above 30 codewords.

Thus we have 226+30 = 256 codewords.

The transmission rule requires that the polarity ofdisparity of adjacent codewords must alternate. In the code table, first 225 input 8 bit words will be assigned balanced words, and rest of the 30 input 8 bit words will be assigned complementary ±2 disparity words. The storage space required is 256x10 bits. For decoding, we can use a EPROM having a suitable size. Now we compare the properties of this code with that proposed by Franaszek.

(a) The maximum run length here = 6. In Franaszek code it was 5

(b) $|RDS|_{max} = 3$ Same as that in that code.

(c) DVS = 6 Same as that in that code.

(d) WRDS = 0, ±2. Same as that in that code.

This is also a DC free code.

Thus there maynot be much advantage in designing 8B/10B code by combining 5B6B and 3B4B codes, as implementation difficulties are expected to be similar in nature.

Since we had discussed in detail the dc contained codes itis of natural interest to discuss the efficiency of such codes. Work in this area has been done by Chien [10]. We discuss the topic in detail below.

4.3 EFFICIENCY OF DC CONSTRAINED CODES

Let H, a positive interger be the desired bound on coded binary signal stream RDS. This defines a subset $s_M(\alpha)$ of all infinite binary sequences An infinite sequence is in the subset $S_M(\alpha)$ if the RDS of the sequence is nowhere larger than M and less than -M. For example

$$\left| \sum_{i=1}^{k} a_i \right| \leq M \quad \text{for } k = 1,2,....$$

A sequence in $S_M(\alpha)$ is an allowable sequece. Denote by $N_M(\alpha)$ and $N(\alpha)$ the number of sequences in $S_M(\alpha)$ and $S(\alpha)$ respectively. Assuming equally probable symbols, the average information per symbol for th sequences in $S_M(\alpha)$ is $\log_2 N_M(\alpha)$.

The efficiency of dc constrained code $\eta = \dfrac{\log_2 N_M(\alpha)}{\log N(\alpha)}$

The next step is to find the number of sequences in $N_M(\alpha)$. Let the sequence length be L. Define an occupancy vector u_L where $u_L = \{u_{-M}, ..., u_o, .., u_{+M}\}$. u_k is the number of allowable sequences of length L with WRDS = k.

The total number of sequences = $N_M(L) = \sum_{k=-M}^{+M} u_k$

As $L \longrightarrow \infty$, $N_M(L) \longrightarrow N_M(\alpha)$ and total number of sequences of length L, $N(L) = 2^L = N(\alpha)$. So $\eta = \lim_{L\to\infty} \frac{\log_2 N_M(\alpha)}{L}$. Further details can be seen in [ ].

In the next chapter, we take a look at the guided scrambling concept and see how it incorporates the features ofthe coding. Finally, we end up with the conclusions on the presentwork with a note on the suggestions for continuation of work in this area.

CHAPTER - 5

PRINCIPLE OF GUIDED SCRAMBLING

Scrambling is a process of randomizing a sequence. This process has the capability to provide balanced sequence and adequate timing information. But, in a worst case, it can also provide least timing information and large run lengths. So, in general, scrambling can produce a sequence which may or may not satisfy line code requirements. So, for line coding purposes we cannot rely totaly on scrambling. Guided scrambling is another technique which involves scrambling but provides an alternative for a sequence for transmission with best line code features.

In this chapter, we briefly discuss the principle of guided scrambling [9].

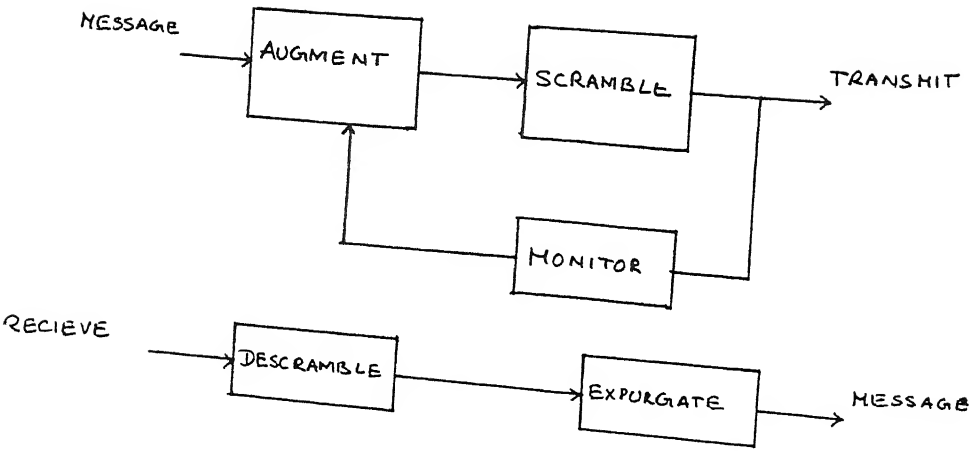
It has been observed that when a sequence is augmented by appending m bits ($m \geq 1$) and then scrambled, for at least one combination of the m bits, a scrambled sequence, which satisfies the line code requirements, is produced. This is the basis for guided scrambling. The principle of guided scrambling looks as shown in Fig. 5.1a. Implementation of a guided scrambling scheme is shown in Fig. 5.1b.

The encoder views the some stream as a series of words k bits in length. Each word is augmented by m bits, resulting in an augmented word of length $n = k+m$. There are 2^m augmenting bit

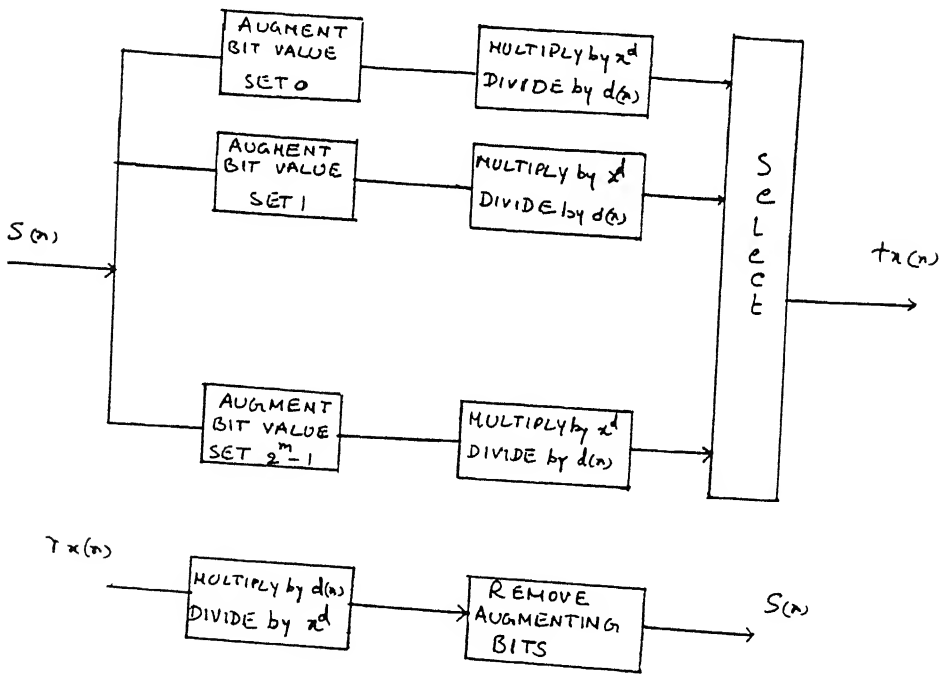
patterns. The resulting augmented bit streams are simultaneously multiplied by x^d and divided by d(x) to form 2^m quotients. The quotient with best line code characteristic is chosen for transmission. Decoding is done through multiplication by d(x) and division by x^d. The augmenting bits are placed in the most significant position. The scrambling polynomial must be properly chosen to ensure that there is at least one quotient with good line code characteristics in every selection set.

The presence of complementary quotient is necessary for balanced transmission. Maximization of the complementary quotients allows for optimisation of secondary line code characteristics. A quotient selection set which consists of 2^m quotients generated by $d(x) = x^m+1$ will contain the maximum of 2^{m-1} pairs of complementary quotients while an appropriate scrambling polynomial ensures a choice for quotient selection, a guided scrambling encoder must select the appropriate quotient to realize the desired output characteristics.

The update of encoding registers can be done by block or continuous or combined block/continuous coding. In block coding, the shift registers are cleared prior to each source word pre-multiplication by x^d and division by d(x). In continuous coding the registers are updated following quotient selection to contain the remainder associated with selected quotient. In combined coding the registers are updated, rather than cleared following quotient selection. In this process, the first d bits of the incoming next augmented word undergo mod2 addition with contents



(a) GS CONCEPT



(b) GS IMPLEMENTATION

Fig 5-1

of register. When $d \leq m$, only augmentation bit values are affected. Decoding through continuous or block multiplication yields same bit stream following removal of augmenting bits.

Description of the diagrams follows :

The message stream is made into blocks of k bits each and to each block, m bits are appended. This takes place in the block "Augment". Scrambling of this augmented bit stream takes place in the block scramble. The scrambled bit stream used for transmission is monitored in the block 'monitor'. Descrambling of the received sequence takes place in the block 'descramble'. The augmented bits are removed in the block 'expurgate'.

A more detailed diagram of GS scheme is shown in Fig.(b). Since there are m augmenting bits, the message stream is augmented by 2^m patterns simultaneously in the blocks $0,1,..2^m-1$. All the 2^m augmented patterns are scrambled in the blocks $x^d/d(x)$. The scrambled versions which best meets the line code requirements is selected in the block 'select'. At the receiving end, the sequence is descrambled in $d(x)/x^d$ block and the augmenting bits are removed to retrieve the message sequence.

CHAPTER - 6

6.1 CONCLUSION

A set of desirable features which the linecode must satisfy in order to provide reliable communication over fibre optic channels was mentioned in the beginning. A detailed study, starting with codes which do not provide error correction and those which have error correcting capability has been done. Primary efforts have been to keep the redundancy as low as possible so that the efficiency of the code is good and the increase in data rate is low.

One important observation is that we have predominantly used only block codes irrespective of whether they are linear or nonlinear, systematic or non-systematic. The reason for this is that, the question is not whether we use a block code or a cyclic code, but the important consideration is that which type of code best meets the line code requirements. We found that block codes are sufficient to meet the requirements and also they are simple to encode and decode. Implementation of these codes also does not pose much difficulties.

Since scrambling is another technique which is also used frequently in digital transmission systems we studied a modification of the same to obtain dc free block codes as presented in chapter five.

In chapter two, we dealt with balanced codes, of both Error detecting and Correcting types. Since, in telecommunication systems, normally 8 bit PCM words are used, these codes having a typical code length of 16 may fit in well. The coset codes presented were derived from partitioning linear block codes. These codes can be easily implemented using ROM tables.

In chapter three, we have proposed some modified block codes which are more efficient than these discussed in chapter two. We calculated the Run length, DSV, RDS etc of these codes, which showed a distinct improvement. A quick comparison of these codes with the dc free coset codes given in chapter two, shows that, for codes without error correction, the maximum run length is n where as for the DC free coset codes, it is $2n + \left[\frac{n}{2}\right]$. The RDS_{max} for these codes is $\frac{n}{2}$, where as for the dc free codes, it is $n+\left[\frac{n}{2}\right]$. A comparison of DC free codes with Error Correction shows that, for these codes, the maximum Run length is n and for DC free codes, it is $2w_{max} + \left[\frac{w_{max}}{2}\right]_J$ where, w_{max} is the maximum weight of a sub-block g_J, $\sum g_J = 1_n$ where n is the code length and 1_n is a all one vector of length n After designing balanced error correcting code, we designed burst correcting balanced array codes. The idea for this design is derived from product codes. We discussed the difference between the product codes and these array codes. One difference is that we transmit the array codeword only row wise. The column codes are not EC codes.The row codes are balanced codes. We discussed the error correction algorithm for the array codes.

In chapter four, we discussed the combined 8B10B code, designed by combining 3B4B and 5B6B codes. This was followed by a directly coded 8B10B code We made a comparison between the two codes above. For combined code, we need three different encoding tables. But, they are smaller in size for the direct code. We need a single table with a size at least 256x10 bits. Later we made a comparison of details like run length, RDS, etc.of the codes.

The maximum Run length for direct code is 6 where as that for combined code is 5. The RDS_{max} for both the codes is 3. The DSV for both the code is 6. The WRDS for boththe codes are 0 or ±2. Both the codes are dc free. We concluded the chapter with a discussion on the efficiency of dc constrained codes.

6.2 SUGGESTIONS FOR FUTURE WORK

We have laid down a theoretical basis for the design of dc free codes. The point of interest is to make an attempt to implement them by hardware means and examine whether or not there are efficient in terms of memory space and speed. Since we have not worked out the power spectral densities of these codes, it would be worthwhile to work out PSd's and try to obtain some information from these details. As mentioned before, we have used only block codes since they are easy to design and meet the requirements. It might be possible to achieve similar results by using the cyclic code approach. One may also take up the convolutional code approach and calculate similar details and make a comparison with the block codes described here, in terms of hardware complexity, ease of implementation, etc.

APPENDIX

DEFINITIONS

(1) Weight : The number of 1's in a codeword.

(2) Hamming distance . The weight of the two codewords x+y where addition is mod-2.

(3) Minimum distance : It is the minimum of the distances between all pairs of codewords. It is represented by d_{min} It has to be at least of in order to detect any error pattern of weight d-1 or less.

(4) A code can detect and correct all patterns of t or fewer errors if and only if the code has $d_{min} \geq 2t+1$.

(5) Disparity : It is defined as the difference between the number of 1's and 0's in a codeword $D = \sum_{i=1}^{n} a_i$.

(6) Running Digital Sum (RDS) : The RDS at any instant is defined as the accumulated difference between the number of transmitted, 1's and 0's in a digital stream.

(7) Word and Running Digital Sum (WRDS) . It is the RDS measured at the end of a codeword.

REFERENCES

[1] Knuth, D.E., "Efficient Balanced Codes", IEEE Transactions on Inf. Theory, Vol. IT-32,No.1, Jan. 1986, pp. 51-53.

[2] Bose, B., "On unordered codes", IEEE Transactions on Computers, Vol 40, No.2, Feb. 1991, pp. 125-131.

[3] Herro, M.A. and Deng, R.H., "DC Free Coset Codes", IEEE Transactions on Inf. Theory, Vol 34, NO. 4, July 1988, pp. 786-792.

[4] Ferreira, "(16,8) DC Free Block Code", IEEE Transactions on Magnetics, Vol. MAG-20, No 5, Sept. 1984.

[5] Balum, "(16,9,6,5,4) EC DC Free Block Code", IEEE Transactions on Inf. Theory, Vol. 34, No. 1, Jan. 1988, pp. 138-142.

[6] Franaszek et al., "ADC Balanced, Partitioned DC Free Block Code", IBM J. RES. DEVELOP, Vol. 27, No. 5, Sept. 1983, pp. 440-451.

[7] Herro, M.A. and L.Hu, "Error Correcting Line Codes", Proc. 23rd Allerton Conf. Comm. Control, Monticello, IL, Oct.1985, pp. 450-451.

[8] Popplewell & O'Reilly, "Spectral Characteristics of a class of DC free ECC", Electronics Letters, 21st July 1988, Vol. 24, No. 15.

[9] Grover et al , "Guided Scrambling : A new line coding technique for high bit rate fibre optic systems", pp.289-296; IEEE Transactions on Comm., Vol. 39, NO. 2, Feb. 1991.

[10] Chien, "Upper bound on the efficiency of DC constrained codes", Bell. Syst. Tech. Journal, Vol. 49, pp. 2267-2287, 1970.

[11] W.W. Peterson, E.J. Weldon Jr., Error Correcting Codes, 2nd ed., Cambridge, MIT Press, 1972.

[12] S. Lin and D.J. Costello Jr., Error Control Coding : Fundamentals and applications, Englewood Cliffs, NJ, Prentice Hall 1983.

[13] TRN Rao, E. Fujiwara, Error Control Coding for Computer Systems, Prentice Hall, 1989.

[14] Sujit Kumar, "Study of Line Codes for Fibre Optic Communications", M.Tech. Thesis, IIT Kanpur, Feb 1992.